



Clase 8. Python

Manejo de archivos y datos

***RECUERDA PONER A GRABAR LA
CLASE***





OBJETIVOS DE LA CLASE

- Conocer el concepto de persistencia
- Procesar datos en .txt
- Elaborar archivos JSON
- Aprender a recuperar datos de fuentes reales (.csv)

CRONOGRAMA DEL CURSO

Clase 7



Métodos de colecciones



COLECCIONES 1



COLECCIONES 2

Clase 8



Manejo de archivos y datos



MI MASCOTA



CURIOSOS POR LA INFORMACIÓN

Clase 9



Funciones



PAR O IMPAR



AÑO BISIESTO

PERSISTENCIA

PERSISTENCIA



Cuando uno piensa en la palabra persistencia, lo relaciona a la capacidad humana de “aguantar”, “perdurar” o “de no dejar de hacer una acción”, en este curso no hablaremos de esas cosas obviamente, pero sí de la **persistencia de los datos.**

PERSISTENCIA

Hasta el momento todos los programas que realizamos ya tenían datos; por ejemplo: las edades de las personas, nombres, apellidos, lista de números, etc.

Pero todos esos datos había que generarlos cada vez que usábamos nuestro programa y los datos se perdían de un día al otro.

La persistencia es lo que nos va a permitir guardar y recuperar los datos que se generaron en algún programa (ya sea nuestro o ajeno).

PERSISTENCIA

En programación, la persistencia es la **acción de preservar la información** de un objeto de forma permanente **(guardado)**, pero a su vez también se refiere a poder recuperar la información del mismo **(leerlo)** para que pueda ser nuevamente utilizado.

De forma sencilla, puede entenderse que **los datos tienen una duración efímera; desde el momento en que estos cambian de valor se considera que no hay persistencia de los mismos.**

TIPOS DE PERSISTENCIA

TIPOS DE PERSISTENCIA

El guardado de datos se puede hacer en dos grandes estructuras.

- En base de datos (o almacenes de datos y sus variantes)



- En archivos



Base de datos

Las bases de datos son sin duda la mejor alternativa para almacenar y explorar los datos. Éstas son complejas y por lo general necesitan de la instalación de algún motor o programa que nos permita utilizarlas.

Trabajaremos con ellas desde la clase 19 en adelante; así que por ahora las



Archivos

Los archivos son la forma más antigua, primitiva y simple de almacenar datos. Pero crearlos o no, aún se sigue utilizando este mecanismo en más de un aplicativo; incluso en programas bastante sofisticados.

En este apartado del curso aprenderemos a **guardar** datos en archivos y **recuperarlos.**

¿Qué son los archivos?

Un archivo o fichero informático es una **secuencia de bytes que son almacenados en un dispositivo**. Un archivo **es identificado por un nombre y la descripción de la carpeta** o directorio que lo contiene. A los archivos informáticos se les llama así porque **son los equivalentes digitales de los archivos escritos** en expedientes, tarjetas, libretas, papel o microfichas del entorno de oficina tradicional.



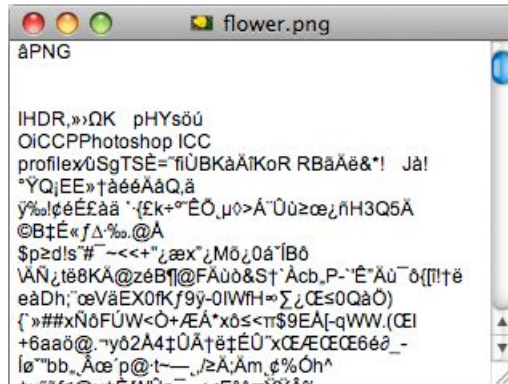
Tipos de archivos



Hay dos grandes tipos de archivos:

- **Los binarios** son aquellos archivos que mejoran su eficiencia pero los datos están guardados bajo agrupaciones de bytes; lo que hace que solo la pc pueda decodificarlos.
- **Los de texto** son lo que uno imagina, texto que guarda el dato en particular de una forma bastante descriptiva e intuitiva.

Ejemplo:



CODER HOUSE

Tipos de archivos



Por simplicidad en el curso trabajaremos solo con archivos de texto. A su vez, los archivos de texto pueden ser de muchas extensiones, las más clásicas son: .txt, .doc, .docx, .xml, .csv, .json, etc.



Escritura de archivos



Escritura de archivos

SOLO porque estamos trabajando con Colabs, debemos realizar lo siguiente:

```
from google.colab import drive
drive.mount('/drive/')
```

Para dar acceso a nuestro drive.

Escritura de archivos



Verificar nuestra identidad:

... Go to this URL in a browser: <https://accounts.google.com/o/oauth2/auth?client>

Enter your authorization code:

Elegir una cuenta

para ir a [Google Drive for desktop](#)

Copia este código, ve a tu aplicación y pégalo en ella:

4/1AX4XfwIBoY40GRsMMfqVU7D0C0-
zdNX8AmBfDrxdulxZVLZHeQkehHmucVM

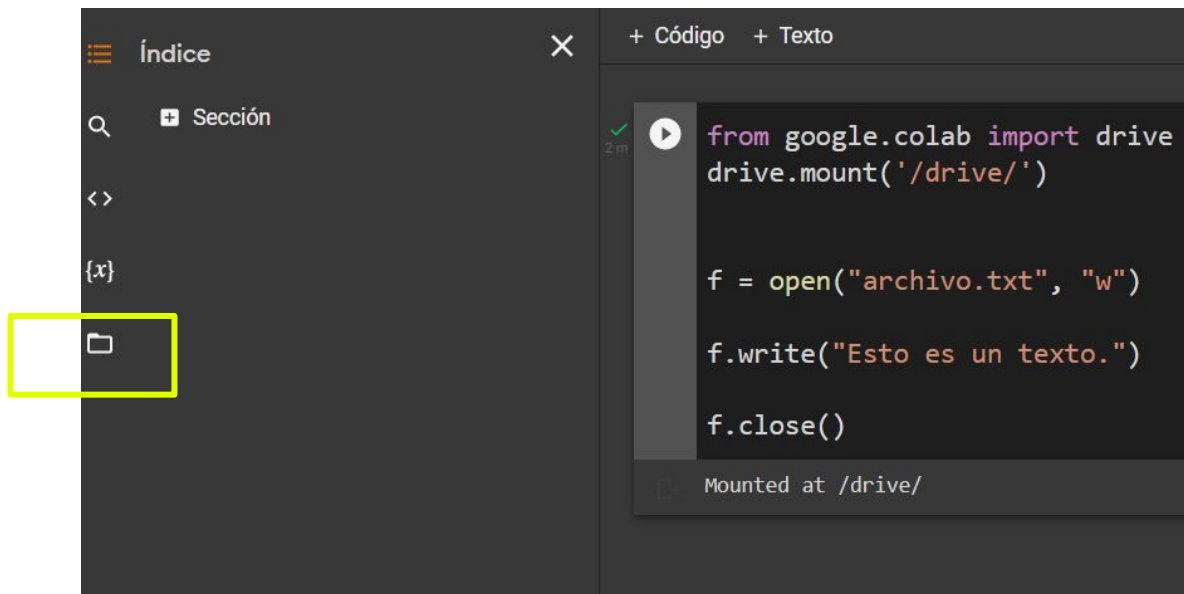


CODER HOUSE

Escritura de archivos



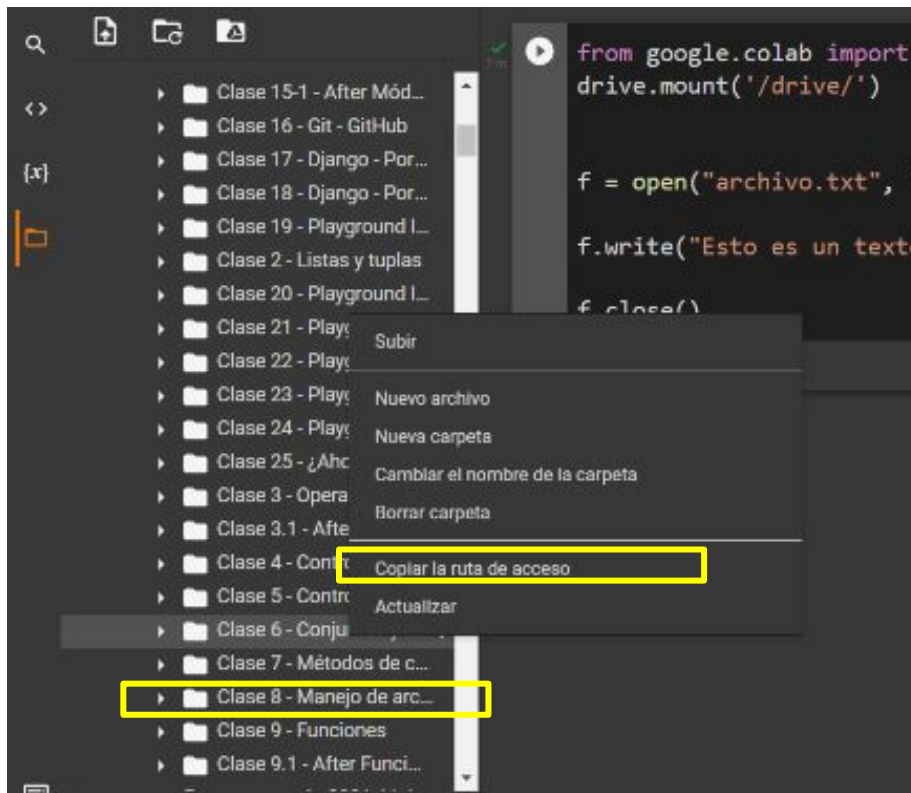
Seleccionar la carpeta donde queremos escribir nuestro archivo:



Escritura de archivos



Seleccionar la carpeta donde queremos escribir nuestro archivo:





Escritura de archivos

Crear una variable con esa ruta:

```
ruta = '/drive/MyDrive/23850-python/Clase 8 - Manejo de archivos y datos'
```

Y luego sólo decidir cómo quieren que se llame el archivo y dar permisos para escribir en él “w”.

Escritura de archivos



```
[3] ruta = '/drive/MyDrive/23850-python/Clase 8 - Manejo de archivos y datos'
```

```
▶ f = open(ruta + "/archivo.txt", "w")  
  
  f.write("Esto es un texto.")  
  
  f.close()
```



archivo.txt 👤



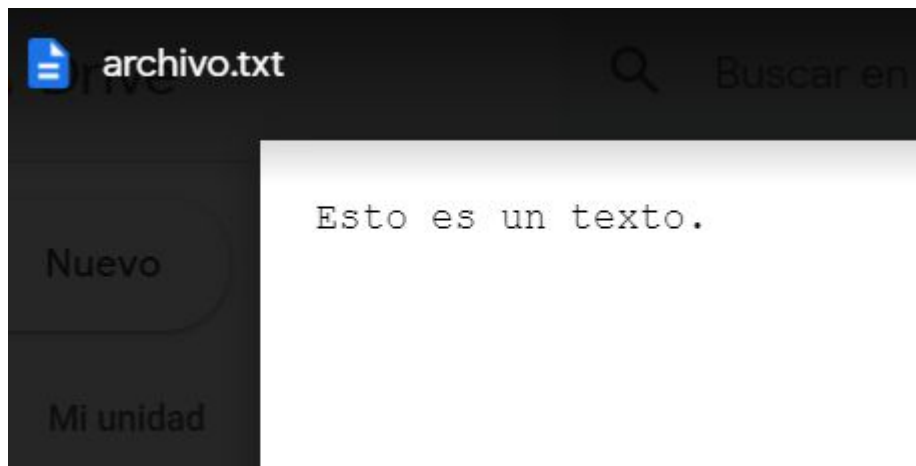
Clase 8 - Manejo de archivos y datos 👤

CODER HOUSE

Escritura de archivos



Dicho archivo quedará así:





Escritura de archivos

Veamos un ejemplo similar pero un poco más completo:

```
nombre = "Nicolas"
apellido = "Perez"
dni = 111111

d = {"NOMBRE":nombre, "APELLIDO":apellido, "DNI":dni}

f = open(ruta + "/otro.txt", "w")

f.write(d["NOMBRE"] + "," + d["APELLIDO"] + "," + str(d["DNI"]) )

f.close()
```




Escritura de archivos

¿Qué creen que se guardó?

Esta es una forma muy útil de guardar información utilizando lo que se llaman delimitadores.

Nombre ↑



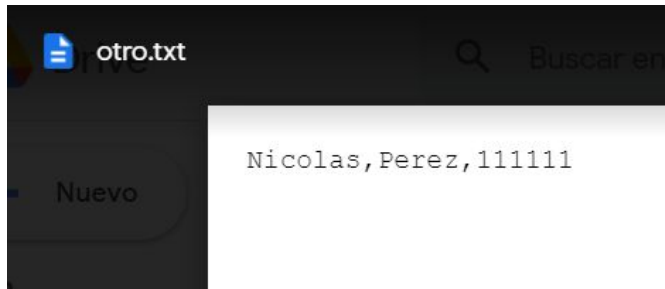
archivo.txt



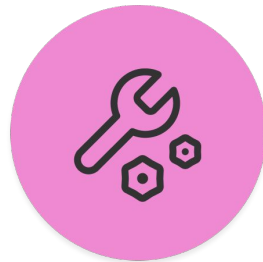
Clase 8 - Manejo de archivos y datos



otro.txt



CODER HOUSE



Mi mascota

Crea un programa y guárdalo en un archivo .txt

Tiempo estimado: 10 minutos

Mi mascota

Desafío
generico



Tiempo estimado: 10 minutos

Crea un programa que pida por teclado (input) los datos de tu mascota y los mismos se guarden en un archivo que se llame miMascota.txt.

EXTRA: Hacerlo con un for o un while para no repetir tanto...!!!



CODER HOUSE

Lectura de archivos

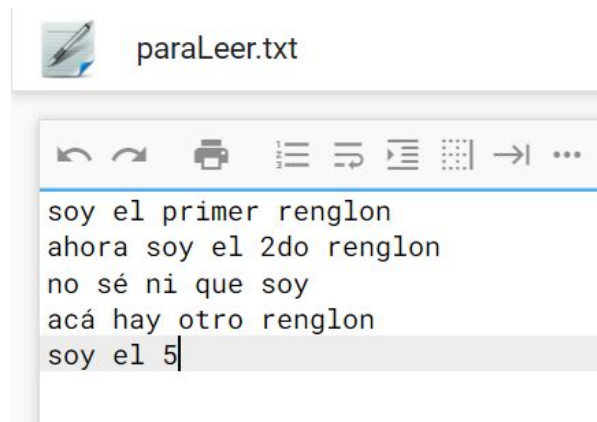
Lectura de archivos

Entonces... ya sabemos inmortalizar nuestros datos en un archivo, pero...
¿cómo haríamos para recuperar esos datos?



Lectura de archivos

Supongamos que queremos recuperar los datos de un archivo cualquiera,
[paraLeer.txt](#)



Lectura de archivos

Es muy sencillo y similar a lo que hicimos con la escritura; solo debemos dar permisos de lectura `(read - r)`.

Además python nos da tres formas de realizarlo:

- `read`

- `readline`

- `readlines`

Lectura de archivos - READ

```
▶ f = open(ruta + "/paraLeer.txt", "r")  
  content = f.read()  
  print(content)  
  f.close()
```

```
↳ soy el primer renglon  
  ahora soy el 2do renglon  
  no sé ni que soy  
  acá hay otro renglon  
  soy el 5
```


Lectura de archivos - READLINE

```
[ ] f = open(ruta + "/paraLeer.txt", "r")  
  
    print(f.readline())  
  
    f.close()  
  
soy el primer renglon
```

**Es lo mismo... ¿Me estas
cargando?**

Solo nos deja ver el primer renglón.

Lectura de archivos - READLINES

```
✓ ▶ f = open(ruta + "/paraLeer.txt", "r")
    for line in f.readlines():
        print(line)

    f.close()
```

☞ soy el primer renglon
ahora soy el 2do renglon
no sé ni que soy
acá hay otro renglon
soy el 5

Es lo mismo... ¿Me estas cargando?

Parece idéntico, pero esto nos
permite acceder renglon por renglon
al txt.

Funcionalidad útil - SEEK

```
✓ [9] f = open(ruta + "/paraLeer.txt", "r")  
0 s  
  
f.seek(20)  
  
print(f.read())  
  
f.close()  
  
n  
ahora soy el 2do renglon  
no sé ni que soy  
acá hay otro renglon  
soy el 5
```

Acceder a una ubicación en particular, es decir, empezar la lectura desde la posición indicada.

¿Qué pasó entonces ahí?, ¿Por qué la n?



BREAK

¡5/10 MINUTOS Y VOLVEMOS!

Archivos JSON

Archivos JSON

Por ahora solo trabajamos con extensiones .txt, pero para el guardado de datos suele ser más útil otro tipo de formato, como el csv, xml o el json.

Archivos JSON - Escritura

```
import json #Importar las funciones de json en la
#clase 15 entenderemos mejor el IMPORT

data = {}

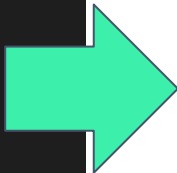
data['clients'] = []

data['clients'].append({
    'first_name': 'Sigrid',
    'last_name': 'Mannock',
    'age': 27,
    'amount': 7.17})

data['clients'].append({
    'first_name': 'Joe',
    'last_name': 'Hinnners',
    'age': 31,
    'amount': [1.90, 5.50]})

data['clients'].append({
    'first_name': 'Theodoric',
    'last_name': 'Rivers',
    'age': 36,
    'amount': 1.11})

with open(ruta + "/primerJson.json", 'w') as file:
    json.dump(data, file, indent=4)
```



```
{
  "clients": [
    {
      "first_name": "Sigrid",
      "last_name": "Mannock",
      "age": 27,
      "amount": 7.17
    },
    {
      "first_name": "Joe",
      "last_name": "Hinnners",
      "age": 31,
      "amount": [
        1.9,
        5.5
      ]
    },
    {
      "first_name": "Theodoric",
      "last_name": "Rivers",
      "age": 36,
      "amount": 1.11
    }
  ]
}
```

CODER HOUSE

Archivos JSON - Lectura

```
[1] with open(ruta + "/primerJson.json") as file:

    dataLectura = json.load(file)

    for client in dataLectura['clients']:
        print('First name:', client['first_name'])
        print('Last name:', client['last_name'])
        print('Age:', client['age'])
        print('Amount:', client['amount'])
        print('')
```

First name: Sigrid
Last name: Mannock
Age: 27
Amount: 7.17

First name: Joe
Last name: Hinners
Age: 31
Amount: [1.9, 5.5]

First name: Theodoric
Last name: Rivers
Age: 36
Amount: 1.11

```
dataLectura

{'clients': [{ 'age': 27,
               'amount': 7.17,
               'first_name': 'Sigrid',
               'last_name': 'Mannock'},
              { 'age': 31,
               'amount': [1.9, 5.5],
               'first_name': 'Joe',
               'last_name': 'Hinners'},
              { 'age': 36,
               'amount': 1.11,
               'first_name': 'Theodoric',
               'last_name': 'Rivers'}]}}
```


Trabajo con datos reales

Trabajo con datos reales - CSV

Para trabajar con datos reales, los mismos pueden estar en cualquier formato, ya sea txt, json o cualquier otro.

Por completitud del curso nosotros supondremos que vienen en csv, así tenemos otro formato más analizado.





Trabajo con datos reales - CSV

En este apartado vamos a leer datos de alguna fuente oficial para poderlos trabajar desde Python.

Para esta clase descargamos datos oficiales de la Nación y de CABA (Argentina).

Datos ciudad de Buenos Aires: <https://data.buenosaires.gob.ar/dataset/>

Datos Nación: <https://datos.gob.ar/>

CODER HOUSE



Trabajo con datos reales - CSV

```
[13] #Con esto daremos permisos para acceder al Drive, nos va a pedir que hagamos login con la cuenta de gmail, y nos dará un código de verificación

from google.colab import drive

#Para generar una variable con los datos de nuestro archivos, vamos a empezar a trabajar con nuestra librería "mágica" para trabajar con datos,
#librería llamada Pandas, para poder usarla debemos hacer lo siguiente

import pandas as pd #En castellano, importamos pandas y lo llamamos pd (pd puede ser cualquier palabra que recuerden)

import numpy as np #Esto lo usaremos bastante más adelante del apunte, por ahora se lo puede ignorar

#En mi caso voy a abrir los datos de turnos de vacunación por covid-19
variableTurnos = pd.read_csv(ruta + "/dataset_turnos_detalle.csv")

#Acá se ve que era necesario tener pandas, ya que usamos la función read_csv de pandas.
#LISTO... Ya tenemos todos nuestros datos leídos y guardados en una variable. Ya son "fácilmente" manipulables.
```

Acá hicimos la lectura de los datos
que nos bajamos

CODER HOUSE

[dataset_turnos_detalle.csv](#)

Trabajo con datos reales - CSV



Admitimos que parece abrumador lo último, pero solo hicimos un read de un dato en otro formato csv. Pero veamos la magia; ya tenemos todos los datos guardados en una variable de Python como para trabajar sin problema:

	genero	sede	servicio	fecha_cita
0	MASCULINO	Centro Islámico	Adultos Mayores de 80 años	01MAR2021:11:15:00
1	MASCULINO	Club Ferro	Adultos Mayores de 80 años	02MAR2021:09:00:00
2	FEMENINO	La Rural	Adultos Mayores de 80 años	01MAR2021:13:30:00
3	MASCULINO	Centro de día Parque Chacabuco	Adultos Mayores de 80 años	25FEB2021:13:00:00
4	FEMENINO	Centro C. Recoleta	Adultos Mayores de 80 años	01MAR2021:13:30:00
...
1925040	FEMENINO	Casa del Historiador	Adultos Mayores de 80 años	23FEB2021:15:45:00

CODER HOUSE

Trabajo con datos reales -CSV



Todo lo que podemos hacer
ahora ya depende de otra rama
de la multifuncionalidad de
Python, que es el análisis de
datos.



Pero veamos lo básico que les
puede servir en este curso:

Trabajo con datos reales -CSV



+ Código + Texto

`#Si esa visualización por defecto no nos gusta podemos verlos de otra forma. Head, nos permite elegir los primeros`

```
variableTurnos.head(3) #Los primeros
```

	genero	sede	servicio	fecha_cita
0	MASCULINO	Centro Islámico	Adultos Mayores de 80 años	01MAR2021:11:15:00
1	MASCULINO	Club Ferro	Adultos Mayores de 80 años	02MAR2021:09:00:00
2	FEMENINO	La Rural	Adultos Mayores de 80 años	01MAR2021:13:30:00

[] `#Con el comando tails, agarramos los ultimos`

```
variableTurnos.tail()
```

	genero	sede	servicio	fecha_cita
1925040	FEMENINO	Casa del Historiador	Adultos Mayores de 80 años	23FEB2021:15:45:00
1925041	MASCULINO	La Rural	Adultos Mayores de 80 años	02MAR2021:15:00:00
1925042	FEMENINO	Club Glorias Argentinas	Adultos Mayores de 80 años	25FEB2021:13:30:00
1925043	MASCULINO	Centro de Dia N9 y 13	Adultos Mayores de 80 años	25FEB2021:08:15:00
1925044	FEMENINO	Club San Lorenzo	Adultos Mayores de 80 años	26FEB2021:11:30:00

[] `#Tambien podemos usar sample, que nos muestra la cantidad que quieras, pero al azar, bastante util.`

```
variableTurnos.sample(3)
```

	genero	sede	servicio	fecha_cita
1404826	FEMENINO	Estadio Luna Park	Adultos Menores de 60 años con condiciones de ...	19JUN2021:11:15:00



Trabajo con datos reales -CSV

```
#Uno de los primeros estadísticos para entrar en tema podría ser las frecuencias simples..  
variableTurnos['sede'].value_counts() #Esto nos muestra cada sede una vez sola, ordenado y con su frecuencia.
```

Club San Lorenzo	264616
La Rural	215748
Estadio Luna Park	196126
Parque Roca	181980
Club River Plate	142751
Usina del Arte	91415
Movistar Arena	82593
Centro Islámico	76093
Centro C. Recoleta	69168
Club Comunicaciones	51846
Club Racing Villa del Parque	51568
Casa del Historiador	31473
PAMI - Centro de Promoción y Prevención	31020
PAMI - Centro de Promoción Prevención y Rehabilitación	30996
PAMI - Agencia 3	30898
Corralón Floresta	29603
Ministerio de Salud	27020
Club Italiano	26348
Centro Cultural El Adan	21973
Club Atlético Boca Juniors	20810

```
variableTurnos["sede"]
```

0	Centro Islámico
1	Club Ferro
2	La Rural
3	Centro de día Parque Chacabuco
4	Centro C. Recoleta
...	
1925040	Casa del Historiador
1925041	La Rural
1925042	Club Glorias Argentinas
1925043	Centro de Dia N9 y 13
1925044	Club San Lorenzo

Name: sede, Length: 1925045, dtype: object

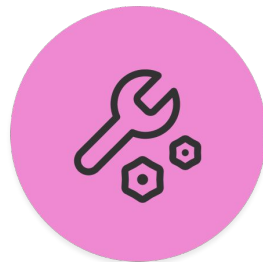


Trabajo con datos reales -CSV

Bueno... Y así podemos seguir todo el día armando y agrupando los datos para manejarlos, cosa que se aleja del cometido del curso, lo importante es que ya saben recuperar datos de tres formatos distintos para reutilizarlos en cualquier proyecto de python... **¿Por qué no, en su proyecto final?**



CODER HOUSE



Curiosos por la información

Tiempo estimado: 15 min



Curiosos por la información

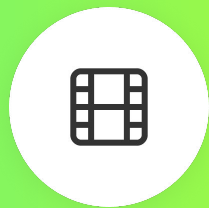
Descargar y guardar en sus Drive algún dato en formato csv de las fuentes que les dimos, o de cualquier otra, leerlos y agruparlos por alguna de las columnas, como hicimos en el ejemplo agrupando por “sede”.

Datos ciudad de Buenos Aires: <https://data.buenosaires.gob.ar/dataset/>

Datos Nación: <https://datos.gob.ar/>

¿PREGUNTAS?





***¿QUIERES SABER MÁS? TE DEJAMOS
MATERIAL AMPLIADO DE LA CLASE***



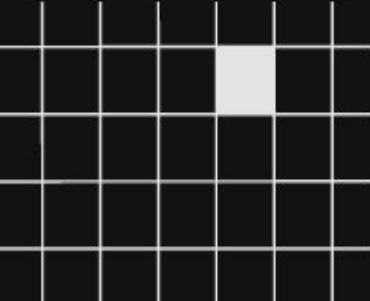
[EjemploClaseEnVivo](#)





¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Lectura de archivos txt, csv y json
 - Escritura de archivos txt y json
 - Manejo de datos - introducción
- 



OPINA Y VALORA ESTA CLASE

#DEMOCRATIZANDO LA EDUCACIÓN

CODER HOUSE